

Limits of quotients of real analytic functions in two variables

Carlos A. Cadavid*, Sergio Molina**, Juan D. Vélez**,

* Corresponding Author, Universidad EAFIT,

Departamento de Ciencias Básicas,

Bloque 38, Office 417. Carrera 49 No. 7 Sur -50,

Medellín, Colombia, ccadavid@eafit.edu.co, (57) (4)-2619500. Fax (57) (4)-3120649.

** Universidad Nacional de Colombia, Departamento de Matemáticas,

Calle 59A No 63 - 20 , Oficina 43-106, Medellín, Colombia,

sdmolina@gmail.com, jdvelez@unal.edu.co

November 9, 2010

Abstract

Necessary and sufficient conditions for the existence of limits of the form

$$\lim_{(x,y) \rightarrow (a,b)} \frac{f(x,y)}{g(x,y)}$$

are given, under the hypothesis that f and g are real analytic functions near the point (a, b) , and g has an isolated zero at (a, b) . An algorithm (implemented in MAPLE 12) is also provided. This algorithm determines the existence of the limit, and computes it in case it exists. It is shown to be more powerful than the one found in the latest versions of MAPLE. The main tools used throughout are Hensel's Lemma and the theory of Puiseux series.

Keywords: Limit, Real Analytic Function, Hensel's Lemma, Puiseaux Series

1 Introduction

In the usual calculus courses one is asked to determine the existence of limits of the form

$$\lim_{(x,y) \rightarrow (a,b)} \frac{f(x,y)}{g(x,y)}$$

where f and g are real analytic functions (typically, polynomials or trigonometric and exponential functions) defined in an open disk centered at a point (a, b) in \mathbb{R}^2 . The standard strategy for solving this problem consists in studying the existence of the limit along various simple trajectories, such as straight lines, quadrics, cubics, etc., with the hope that either one of them fails to exist or two of them differ. If they all coincide, then one tries some other ad hoc trajectories. If all that fails, one tries to prove its existence by some theoretical methods.

In this paper we develop a theoretical method which completely solves this problem. An algorithm for polynomials based on this method is implemented, which proves to be more powerful than other existing routines.

An application of Weierstrass' Preparation Theorem allows to reduce the problem to the case where f and g are monic polynomial functions in the variable y , whose coefficients are real series in the variable x . Next, a *discriminant real curve* is constructed using Lagrange Multipliers with the property that the limit exists, if and only if, it exists along this curve. Then Hensel's lemma, some Galois theory, and the theory of Puiseaux series are used to parametrize the various branches of the discriminant curve and select the *real ones*. All these steps are done in a constructive manner making it possible to implement this method in an algorithmic way. In this article an algorithm was implemented for polynomial functions.

2 Theory

2.1 Reduction to the case where f and g are polynomials

After a translation, we may assume that (a, b) is the origin.

Let us denote by $S = \mathbb{R}\{x, y\}$ the ring of power series in the variables x, y with real coefficients having positive radius of convergence around the

origin. If $h(x, y)$ belongs to S , the order of h in the variable y is defined to be the smallest integer r such that $\bar{h}(y) = h(0, y)$ has the form

$$\bar{h}(y) = \alpha_r y^r + \alpha_{r+1} y^{r+1} + \cdots, \text{ with } \alpha_r \neq 0.$$

(If $\bar{h}(y) = 0$ the order is defined to be $+\infty$.)

It is not difficult to show that given h_1, \dots, h_n in $S - \{0\}$ there exists an integer $v \geq 1$ such that after a change of coordinates of the form $x' = x + y^v, y' = y$, each series $h'_i(x', y') = h_i(x + y^v, y)$ is of finite order in the variable y' [GLS]. The essential tool for the reduction is the following lemma.

Lemma 1 (Weierstrass) *Let h be an element of $S = \mathbb{R}\{x, y\}$ of order d in y . Then there exists a unique unit $u(x, y) \in S$ and unique real series $a_1(x), \dots, a_d(x)$ with positive radii of convergence such that $h(x, y) = u(x, y)(y^d + a_1(x)y^{d-1} + \cdots + a_d(x))$ [GLS].*

Since the existence of the limit and its value is obviously independent of the particular choice of local coordinates, we may assume that $f(x, y) = u(x, y)f_1(x, y)$ and $g(x, y) = v(x, y)g_1(x, y)$, where $u(x, y)$ and $v(x, y)$ are units and

$$\begin{aligned} f_1 &= y^d + a_1(x)y^{d-1} + \cdots + a_d(x) \\ g_1 &= y^b + c_1(x)y^{b-1} + \cdots + c_b(x) \end{aligned}$$

are monic polynomials in $\mathbb{R}\{x\}[y]$. Since units do not affect the existence of the limit, there is no loss of generality in assuming also that u and v are equal to 1.

2.2 Discriminant variety for the limit

The following proposition provides a necessary and sufficient condition for the existence of the limit.

Let $f = y^d + a_1(x)y^{d-1} + \cdots + a_d(x)$ and $g = y^b + c_1(x)y^{b-1} + \cdots + c_b(x)$ be monic polynomials in $\mathbb{R}\{x\}[y]$, and $D = D_\rho(0) \subset \mathbb{R}^2$ a closed disk centered at the origin with radius $\rho > 0$, such that each $a_i(x), c_j(x)$ is convergent in D . Let us denote by h' the polynomial $y\partial q/\partial x - x\partial q/\partial y$, where q denotes the quotient $q = f/g$, and by h'' (the numerator of h') the polynomial

$$h'' = y \left(g \frac{\partial f}{\partial x} - f \frac{\partial g}{\partial x} \right) - x \left(g \frac{\partial f}{\partial y} - f \frac{\partial g}{\partial y} \right). \quad (1)$$

Let X be the variety cut by h' in the puncture disk, i.e.,

$$X = \{(x, y) \in D : (x, y) \neq (0, 0) \text{ and } h''(x, y) = 0\}.$$

With this notation we have the following proposition.

Proposition 2 *Let $q(x, y) = f(x, y)/g(x, y)$. The limit*

$$\lim_{(x,y) \rightarrow (a,b)} q(x, y)$$

exists and equals $L \in \mathbb{R}$, if and only if for every $\epsilon > 0$ there is $0 < \delta < \rho$ such that for every $(x, y) \in X \cap D_\delta$, the inequality $|q(x, y) - L| < \epsilon$ holds.

Proof. The method of Lagrange multipliers applied to the function $q(x, y)$, subject to the condition $x^2 + y^2 = r^2$ where $0 < r < \rho$, says that the extreme values taken by $q(x, y)$ on each circle $C_r(0)$, centered at the origin and having radius r , occur among those points (x, y) of $C_r(0)$ for which the vectors $(\partial q/\partial x, \partial q/\partial y)$ and (x, y) are parallel, which amounts to $y\partial q/\partial x - x\partial q/\partial y = 0$. Let us assume that given $\epsilon > 0$ there exists $0 < \delta < \rho$ such that for every $(x, y) \in X \cap D_\delta$ the inequality $|q(x, y) - L| < \epsilon$ holds. Let $(x, y) \in D_\delta$ and $r = \sqrt{x^2 + y^2}$. If $t_1(r), t_2(r) \in C_r(0)$ are such that $q(t_1(r)) = \min_{t \in C_r(0)} q(t)$, and $q(t_2(r)) = \max_{t \in C_r(0)} q(t)$, then

$$q(t_1(r)) - L \leq q(x, y) - L \leq q(t_2(r)) - L,$$

for every $(x, y) \in C_r(0)$. Since $t_1(r), t_2(r) \in X \cap D_\delta$ we have

$$-\epsilon < q(t_1(r)) - L \text{ and } q(t_2(r)) - L < \epsilon$$

and therefore $|q(x, y) - L| < \epsilon$, for all $(x, y) \in D_\delta$.

The "only if" part is immediate from the definition of limit. ■

2.3 Hensel's Lemma

Let us fix an integer $n \geq 0$. A linear change of coordinates of the form $f_1(x, y) = f(x+ny, -nx+y)$, and $g_1(x, y) = g(x+ny, -nx+y)$ does not alter the limit of the quotient as (x, y) approaches the origin. It is easy to see that this change of coordinate transforms (1) into a monic polynomial multiplied

by a nonzero constant. By the chain rule we have that $h''(x + ny, -nx + y)$ is equal to

$$(-nx + y)[g_1(\partial f/\partial x)_1 - f_1(\partial g/\partial x)_1] - (x + ny)[g_1(\partial f/\partial y)_1 - f_1(\partial g/\partial y)_1],$$

where $(\partial f/\partial x)_1(x, y) = \partial f/\partial x(x + ny, -nx + y)$, and similarly with $(\partial f/\partial y)_1, (\partial g/\partial x)_1, (\partial g/\partial y)_1$. We will denote $h''(x + ny, -nx + y)$ simply by $h(x, y)$.

Our next goal is to parameterize the curve $h(x, y) = 0$. For this purpose we will use Hensel's Lemma ([E]). Let us denote by k an arbitrary field, by R the ring of formal power series in the variable x , with coefficients in k , $R = k[[x]]$, and by $k((x))$ its field of fractions. R is a local ring (R, m) whose maximal ideal is $m = (x)$. For each $h(x, y) \in R[y]$ monic in the variable y , let us denote by \bar{h} its reduction modulo m , i.e., $\bar{h} = h(0, y)$.

Lemma 3 (Hensel's Lemma) *Let $F(x, y)$ be an element of $R[y]$ monic in y , and let us assume that $\bar{F} = gh$ is a factorization in $k[y]$ whose factors are relatively prime, and of degrees r and s . Then there exist unique G and H in $R[y]$ with degrees r and s , respectively, such that:*

$$1. \bar{G} = g \text{ and } \bar{H} = h$$

$$2. F = GH$$

In order to construct a parameterization of $h(x, y) = 0$, Puiseaux series are used which we review next.

2.4 Puiseaux Series

Let us denote by L the quotient field of fractions of $R = \mathbb{C}[[x]]$, which consists of Laurent series. Let \bar{L} be an algebraic closure of L . For each positive integer n we will denote by $x^{1/n}$ a fixed n -th root of x in \bar{L} . It is clear that the n -th roots of x are

$$\theta x^{1/n}, \theta^2 x^{1/n}, \dots, \theta^{n-1} x^{1/n}, x^{1/n}$$

where θ is any primitive n -th root of unity. It is easy to see that the polynomial $t^n - x$ is irreducible in $L[t]$ and therefore $L \subset L(x^{1/n})$ is an extension of degree n . Consider the directed system consisting of the positive natural numbers (partially) ordered by divisibility, i.e., $n \leq m$ if and only if $n|m$. The direct limit $\lim_{\rightarrow \mathbb{N}} L(x^{1/n})$ will be denoted by L^* . This limit can be identified with the field $\cup_n L(x^{1/n}) \subset \bar{L}$. Each element σ of L^* can therefore be written in the form $\sigma = \sum c_k x^{q_k}$, with $c_k \in \mathbb{C}$, and exponents $q_k \in \mathbb{Q}$ such that:

1. $q_1 < \cdots < q_r < \cdots$
2. There is an integer b so that each exponent can be written as $q_i = a_i/b$, for some integer a_i .

The least exponent in the expression for σ , q_1 , is called *the order of σ* . It is a well known theorem that given a monic polynomial

$$h(x, y) = y^d + h_1(x)y^{d-1} + \cdots + h_d(x)$$

in $R[y]$, there is an integer $N > 0$ such that h can be factored completely in $L(x^{1/N}) \subset L^*$ as

$$h(x, y) = (y - \sigma_1(x^{1/N})) \cdots (y - \sigma_d(x^{1/N})), \quad (1)$$

where each $\sigma_i(t)$ is an element of $\mathbb{C}[[t]]$, i.e. a formal power series. Moreover, it can be seen that this series has positive radius of convergence and therefore defines a holomorphic function (cf. [GLS]). Using this result, it is possible to *parameterize* the curve

$$X = \{(x, y) \in \mathbb{C}^2 : (x, y) \neq (0, 0) \text{ and } h(x, y) = 0\}.$$

The proof of the existence of (1) can be done constructively using Hensel's Lemma (Lemma 8) making it possible to determine which one of the series $\sigma_i(t)$ has only real coefficients. Such series will be called throughout, a *real series*. This in turn allows us to parameterize each one of the trajectories in $X \cap \mathbb{R}^2$ which go through the origin. It will be shown that these are the only ones that are relevant, since for a holomorphic function to be real valued on a real sequence approaching zero, it must have a series expansion around the origin with only real coefficient.

The parameterization of the zeroes of h can be done by observing that

$$h(x^N, y) = (y - \sigma_1(x)) \cdots (y - \sigma_d(x))$$

and consequently $X = \{(x, y) \in \mathbb{C}^2 : h(x, y) = 0\}$ is the union of the sets $X_i = \{(z^N, \sigma_i(z)) : z \in \mathbb{C}\}$. This allows us to prove the following central result.

Theorem 4 *Let $\sigma_1(z), \dots, \sigma_l(z)$, $l \leq d$, be the real series in the equation (1) which go through the origin (i.e. $\sigma_i(0) = 0$, for $i = 1, \dots, l$) . Then the limit*

$$\lim_{(x,y) \rightarrow (0,0)} \frac{f_1(x, y)}{g_1(x, y)}$$

exists if and only if

$$\lim_{t \rightarrow 0} \frac{f_1(t^N, \sigma_i(t))}{g_1(t^N, \sigma_i(t))} = L_i$$

exists, for $i = 1, \dots, l$, and $L_1 = \dots = L_l$.

2.5 Newton's automorphism

For each rational number $q \neq 0$ there exists a homomorphism $\alpha_q : L^* \rightarrow L^*$, which sends x to x^q and fixes the subfield \mathbb{C} . This homomorphism is constructed by first defining a homomorphism from $\mathbb{C}[x]$ into L^* which sends x to x^q , then extending it to $\mathbb{C}[[x]]$, and then to the field of fractions $\mathbb{C}((x))$. Since L^* is an algebraic extension of L , this homomorphism extends to a homomorphism α_q from L^* to \overline{L} . It is clear that the image α_q lies inside L^* , and therefore one can regard α_q as an endomorphism of L^* . For $p \neq 0$ rational, let $\beta_{q,p} : L^*[y] \rightarrow L^*[y]$ be the extension of α_q obtained sending y to yx^p . It is clear that $\beta_{q,p}$ is invertible and its inverse is $\beta_{1/q, -p/q}$.

With these preliminaries we can now state the following fundamental theorem ([M]). Even though this result is well known in the literature, we provide a "constructive" proof, since it is the very heart of the procedure *sus* in the algorithm *limite*, whose code we give at the end.

Theorem 5 *Every polynomial $h = y^d + h_1(x)y^{d-1} + \dots + h_d(x)$ with coefficients $h_i(x)$ in L^* can be factored into linear factors $h = (y - \sigma_1) \cdots (y - \sigma_d)$, with $\sigma_i \in L^*$. Even more, if each $h_i(x)$ belongs to $\mathbb{C}[[x]]$, then there exists a positive integer n such that all $\sigma_i \in \mathbb{C}[[x^{1/n}]]$.*

Proof. The proof proceeds by induction on the degree of h . The homomorphism $\phi : L^*[y] \rightarrow L^*[y]$ sending y to $y - (h_1(x)/d)$, and fixing every element of L^* , is invertible and its inverse is the homomorphism that fixes each element of L^* and sends y to $y + (h_1(x)/d)$. A simple calculation shows that $\phi(h)$ is a polynomial with coefficients in L^* such that the coefficient of y^{d-1} is zero. Then

$$\phi(h) = y^d + b_2(x)y^{d-2} + \dots + b_d(x).$$

Let us denote by u_i the order of $b_i(x)$, and let $2 \leq r \leq d$ be the least index for which $u_r/r = \min\{u_i/i : 2 \leq i \leq d\}$. Let us define $\psi = \beta_{r,u_r} : L^*[y] \rightarrow L^*[y]$.

ψ sends x to x^r , y to $x^{u_r}y$ (and its inverse sends x to $x^{1/r}$, and y to $x^{-u_r/r}y$).

Clearly,

$$\psi(\phi(h)) = x^{du_r}y^d + b_2(x^r)x^{(d-2)u_r}y^{d-2} + \cdots + b_d(x^r) \quad (2)$$

$$= x^{du_r}(y^d + x^{-2u_r}b_2(x^r)y^{d-2} + \cdots + x^{-ku_r}b_k(x^r)y^{d-k} + \cdots + x^{-du_r}b_d(x^r)). \quad (3)$$

The order of each term $x^{-ku_r}b_k(x^r)$ of the polynomial inside the parentheses is given by $-ku_r + ru_k \geq 0$ (since $u_k/k \geq u_r/r$). Furthermore, the order of $x^{-ru_r}b_r(x^r)$ is 0. Therefore, if

$$F' = y^d + x^{-2u_r}b_2(x^r)y^{d-2} + \cdots + x^{-ku_r}b_k(x^r)y^{d-k} + \cdots + x^{-du_r}b_d(x^r),$$

by taking N large enough, we have that $F = F'(x^N, y) \in \mathbb{C}[[x]][y]$, and F admits a modulo x reduction $f = \overline{F} \in \mathbb{C}[y]$ having at least two distinct roots. For if f had a single root c , then it is impossible that $c = 0$ because the order of $x^{-ru_r}b_r(x^r)$ is zero. And if $c \neq 0$ then $f = (y - c)^d = y^d - dcy^{d-1} + \cdots$ would have a nontrivial term in y^{d-1} , which is also impossible. Therefore, since $f \in \mathbb{C}[y]$, one obtains $f = f_1f_2$, with f_1 and f_2 monic and of degrees strictly smaller than d . Hensel's Lemma guarantees the existence of a lifting $F = F_1F_2$ with F_1 and F_2 monic, and of degrees strictly smaller than the degree of F . Consequently, $F'(x, y) = F'_1F'_2$ with $F'_i = F_i(x^{1/N}, y)$. In conclusion, $\psi(\phi(h)) = x^{du_r}F'_1F'_2$. By the induction hypothesis we know that $F'_1F'_2 = \prod_{j=1}^d(y - \sigma_j)$, with $\sigma_j \in L^*$ and therefore

$$\phi(h) = x^{du_r/r}\psi^{-1}(F'_1F'_2) \quad (4)$$

$$= x^{du_r/r} \prod_{j=1}^d(x^{-u_r/r}y - \psi^{-1}(\sigma_j)) \quad (5)$$

$$= \prod_{j=1}^d(y - x^{u_r/r}\psi^{-1}(\sigma_j)) \quad (6)$$

is also a product of linear factors. This finishes the induction. The last claim in the theorem also follows by induction on the degree of h . In fact, if the coefficients of h belong to $\mathbb{C}[[x]]$, then $F' \in \mathbb{C}[[x]][y]$ and consequently F'_1 and F'_2 also belong to $\mathbb{C}[[x]][y]$. The induction hypothesis guarantees the existence of a positive integer m such that $F'_1F'_2 = \prod_{j=1}^d(y - \sigma_j)$, with $\sigma_j \in \mathbb{C}[[x^{1/m}]]$. Furthermore, in the polynomial $\phi(h)$ all the u_k are nonnegative and therefore

$u_r/r \geq 0$. Consequently, each element $x^{u_r/r}\psi^{-1}(\sigma_j) = x^{u_r/r}\sigma_j(x^{1/r})$ belongs to $\mathbb{C}[[x^{1/n}]]$ with $n = rm$, and therefore the desired factorization for h is obtained. ■

It easily follows from the last part of the proof of theorem 5 shows that

Remark 6 *If $h = y^d + h_1(x)y^{d-1} + \cdots + h_d(x)$ is a monic polynomial with coefficients in $\mathbb{C}[[x]]$, then there exists a power $r > 0$, and polynomials $g_1(x, y)$ and $g_2(x, y)$ in $\mathbb{C}[[x]][y]$, which are monic in the variable y and of degrees $d_1, d_2 < d$, such that $h(x^r, y) = g_1(x, y)g_2(x, y)$.*

Theorem 5 admits the following refinement ([M]).

Theorem 7 *Let $h = y^d + h_1(x)y^{d-1} + \cdots + h_d(x)$ be a monic polynomial with coefficients in $\mathbb{C}[[x]]$ that is irreducible over $L = \mathbb{C}((x))[y]$. Then, if ω denotes a primitive d^{th} root of unity, there exists $\sigma(t) = \sum_{k=0}^{\infty} c_k t^k$ such that*

$$h = (y - \sigma(\omega x^{1/d})) \cdots (y - \sigma(\omega^d x^{1/d}))$$

where $\sigma(\omega^r x^{1/d}) = \sum_{k=0}^{\infty} c_k (\omega^r x^{1/d})^k$.

The following result allows to identify the real series in the factorization given in (5).

Lemma 8 *Let $F = y^d + b_1(x)y^{d-1} + \cdots + b_d(x)$ be a monic polynomial of degree d in the variable y and whose coefficients are real power series, i.e. $b_i(x) \in \mathbb{R}[[x]]$. Then $\overline{F} = (y - r)^d$, with $r \in \mathbb{R}$, if and only if its factorization in Puiseaux series has the form*

$$F = \prod_{i=1}^s F_i \tag{7}$$

where

$$F_i = (y - \sigma_i(x^{1/d_i}))(y - \sigma_i(\omega_i x^{1/d_i})) \cdots (y - \sigma_i(\omega_i^{d_i-1} x^{1/d_i})),$$

and each $\sigma_i(t) = \sum_{k=0}^{\infty} c_k t^k$ is a real series, ω_i is a primitive d_i^{th} root of unity, $\sum_{i=1}^s d_i = d$, and $\sigma_i(0) = r$.

Proof. Let us first consider the only if part of the equivalence. We proceed by induction on d . If the degree of F is $d = 1$, then $F = y - b_1(x)$, with $b_1(x) \in \mathbb{R}[[x]]$, is a factorization in Puiseaux series and $b_1(0) = r$. If $d > 1$, by remark 6, there exists an integer $N > 0$ such that $F(x^N, y) = G(x, y)H(x, y)$ where G and H are polynomials which are monic and of degrees $d_1, d_2 < d$ in the variable y . Thus

$$\overline{F} = F(0, y) = G(0, y)H(0, y).$$

And this implies that

$$\overline{G} = (y - r)^{d_1}, \text{ and } \overline{H} = (y - r)^{d_2}.$$

By Hensel's Lemma applied to the ring $\mathbb{R}[[x]][y]$, and in particular, by its claim about uniqueness, one obtains that G and H are polynomials with coefficients in $\mathbb{R}[[x]]$. By the induction hypothesis G and H can be factored in the form (7), so F can also be factored in this way. Conversely, if (7) holds, then it follows, by taking $N = d_1 \dots d_s$, that

$$F(x^N, y) = \prod_{i=1}^s (y - \sigma_i(x^{e_i}))(y - \sigma_i(\omega_i x^{e_i})) \cdots (y - \sigma_i(\omega_i^{d_i-1} x^{e_i})) \quad (8)$$

where $e_i = N/d_i$ and $\sigma_i(t)$ is a real series. By replacing x by 0 in (8) one obtains

$$\overline{F} = F(0, y) = \prod_{i=1}^s (y - \sigma_i(0))^{d_i} = (y - r)^d.$$

■

Let now $F = y^d + b_1(x)y^{d-1} + \cdots + b_d(x)$ be a monic polynomial whose coefficients are in $\mathbb{R}[[x]]$ and let us denote its reduction modulo x by f . We can write

$$f = (y - r_1)^{d'_1} \cdots (y - r_s)^{d'_s} (y - c_1)^{d_1} (y - \overline{c_1})^{d_1} \cdots (y - c_l)^{d_l} (y - \overline{c_l})^{d_l},$$

where r_1, \dots, r_s are the real roots of f , and $c_i, \overline{c_i}$ are the nonreal ones. Let us define $f_i(y) = (y - r_i)^{d'_i}$ and

$$g_i(y) = (y - c_i)^{d_i} (y - \overline{c_i})^{d_i} = (y^2 - \alpha_i y + \beta_i)^{d_i},$$

with α_i, β_i real. Hensel's Lemma provides us with a lifting of the factorization $f_1 \cdots f_s g_1 \cdots g_l$, of the form

$$F = F_1 \cdots F_s G_1 \cdots G_l$$

i.e., $\overline{F_i} = f_i$ and $\overline{G_i} = g_i$. From the proof of Hensel's Lemma it follows that each F_i is a monic polynomial in the variable y with coefficients in $\mathbb{R}[[x]]$. Each G_i admits a factorization $\prod_{j=1}^{q_i} G_{ij}$ into irreducible factors in $\mathbb{C}[[x]][y]$, and, by Gauss' Lemma, also in $\mathbb{C}((x))[y]$ (see [L]). Notice that if G_{ij} has degree e_{ij} , then $\sum_{j=1}^{q_i} e_{ij} = 2d_i$. By theorem 7 each G_{ij} can be factored as

$$G_{ij} = \prod_{k=1}^{e_{ij}} (y - \sigma_j(\omega_j^k x^{1/e_{ij}})),$$

where ω_j is an e_{ij}^{th} primitive root of unity and $\sigma_j(t) \in \mathbb{C}[[t]]$. If we let $e_i = e_{i1} \dots e_{iq_i}$, it is clear that

$$G_i(x^{e_i}, y) = \prod_{j=1}^{q_i} \prod_{k=1}^{e_{ij}} (y - \sigma_j(\omega_j^k x^{e_i/e_{ij}})),$$

and as a consequence

$$G_i(0, y) = \prod_{j=1}^{q_i} (y - \sigma_j(0))^{e_{ij}}.$$

It follows that $q_i = 2$ and $\sigma_1(0) = c_i$ and $\sigma_2(0) = \bar{c}_i$ (or the other way round). Therefore none of the σ_j is a real series. By lemma 8, $F_k = \prod_{h=1}^m F_{kh}$, and

$$F_{kh} = (y - \sigma_{hk}(x^{1/d_{hk}}))(y - \sigma_{hk}(\omega_{hk} x^{1/d_{hk}})) \cdots (y - \sigma_{hk}(\omega_{hk}^{d_{hk}-1} x^{1/d_{hk}})), \quad (9)$$

with $\sigma_{hk}(t)$ being a real series. The following theorem summarizes what has been achieved so far.

Theorem 9 *Let $F = y^d + b_1(x)y^{d-1} + \cdots + b_d(x)$ be a polynomial that is monic in the variable y and whose coefficients lie in $\mathbb{R}[[x]]$, and let f be its reduction modulo x . Then f can be written as*

$$f = (y - r_1)^{d'_1} \cdots (y - r_s)^{d'_s} (y - c_1)^{d_1} (y - \bar{c}_1)^{d_1} \cdots (y - c_l)^{d_l} (y - \bar{c}_l)^{d_l}.$$

Let

$$f_i(y) = (y - r_i)^{d'_i}, \quad g_i(y) = (y - c_i)^{d_i} (y - \bar{c}_i)^{d_i} = (y^2 - \alpha_i y + \beta_i)^{d_i},$$

with r_i, α_i, β_i real. Hensel's Lemma gives a lifting of the factorization of $f = f_1 \cdots f_s g_1 \cdots g_l$

$$F = F_1 \cdots F_s G_1 \cdots G_l$$

with $\overline{F_i} = f_i$ and $\overline{G_i} = g_i$. Then, in the Puiseaux series factorization of F the only real series occur in the decomposition into linear factors of the F_i .

Thus if X denotes the curve in \mathbb{C}^2 formed by the zeroes of h then

$$X \cap \mathbb{R}^2 = \cup_{i=1}^s \{(t^d, \sigma_i(t^{m_i})) : t \in [a, b] \subset \mathbb{R}\}.$$

3 Algorithm for the computation of limits

As an application of the results established in the previous section, we present in this section an algorithm implemented in *Maple 12* for the determination of limits of the form

$$\lim_{(x,y) \rightarrow (0,0)} \frac{f(x,y)}{g(x,y)}$$

where $f, g \in \mathbb{R}[x, y]$ and g has an isolated zero at $(0, 0)$. The complete routine is called *limite* and comprises several subroutines that are documented next.

The routine *repeticion* takes as input a list L and forms a list of lists, each one formed by each element of L , repeated as many times as it appears in L . For instance, if $L = [1, 2, 1, 3, 1, 2, 4, 3]$ then *repeticion* produces the list $[[1, 1, 1], [2, 2], [3, 3], [4]]$. The *Maple 12* code for this routine is as follows.

```
> repeticion:=proc(L)
> local S,j,H,i;
> H:=convert(convert(L,set),list);
> for i from 1 to nops(H) do S[i]:=[];
> for j from 1 to nops(L)do
> if L[j]=H[i] then S[i]:=[op(S[i]),H[i]]; end if;
> end do; end do;
> RETURN([seq(S[i], i=1..nops(H))]);
> end proc;
```

The routine *suprime* takes as input a list L and eliminates its redundancy. For instance, if $L = [1, a, 1, 3, a, b, c]$, then *suprime* returns $G = [1, a, 3, b, c]$. The code for this routine is

```
> suprime:=proc(L)
> local S,i,G; G:=[L[1]];
> for i from 2 to nops(L) do
> S:=convert(evalf(G),set);
> if (evalb(`in`(evalf(L[i]),S)) = false) then G:=[op(G),L[i]];
> end if;
> end do;
```

```
> RETURN(G);
> end proc:
```

The routine *poli* takes as input a list L whose elements are complex numbers and constructs another list containing a polynomial of the form $(y - r)^d$ for each real r that appears exactly d times in L , and a polynomial of the form $((y - z)(y - \bar{z}))^d$ for each nonreal z appearing together with its conjugate \bar{z} exactly d times in L . For example, if $L = [1, 2 - i, 1, 2 + i]$ then *poli* returns $[(y - 1)^2, (y - (2 - i))(y - (2 + i))]$. Its code is

```
> poli:=proc(L)
> local g,H,i;
> H:=repeticion(L);
> for i from 1 to nops(H) do
> if Im(evalf(H[i][1]))=0 then g[i]:=(y-H[i][1])^nops(H[i]); else
> g[i]:=((y-H[i][1])*(y-conjugate(H[i][1])))^nops(H[i]); end if; end do;
> suprime([seq(g[i],i=1..nops(H))]);
> end proc:
```

The routine *mochar* takes a polynomial $f(x, y) = a_0(x)y^d + \dots + a_k(x)y^k + \dots + a_d(x)$ and eliminates from each coefficient those powers of x which are larger than n , i.e. it calculates $f(x, y)$ modulo x^{n+1} . The code for this routine is

```
> mochar:=proc(f,n)
> local c,d,i,g;
> d:=degree(f,y); g:=0;
> for i from 0 to d do
> c[d-i]:=mtaylor(coeff(f,y,d-i), [x], n+1);
> g:=g+c[d-i]*y^(d-i);
> end do;
> collect(g,y);
> end proc:
```

The routine *monico* takes a polynomial f in the variable y and divides it by the coefficient of the highest power of y . Its code is

```
> monico:=proc(f)
> local c,d;
> d:=degree(f,y);
> c:=coeff(f,y,d);
```

```

> collect(expand(1/c*f),y);
> end proc:
```

The routine *Hensel* has four entries. The first entry is a polynomial $F(x, y)$ en $\mathbb{C}[x][y]$, which is monic in y . The second and third entries are polynomials $g(x)$, $h(y)$ such that $F(0, y) = g(y)h(y)$. The fourth entry is an integer n . *Hensel* calculates polynomials $G(x, y)$ and $H(x, y)$ such that $F = GH$ modulo x^{n+1} . The code for this routine is

```

> Hensel:=proc(poly,gg,hh,n)
> local L,H,G,i,l,f,g,h,t;
> f[0]:=coeff(poly,x,0);
> g[0]:=gg;
> h[0]:=hh;
> f[1]:=coeff(poly,x,1);
> gcdex(g[0],h[0],f[1],y,'s','t'); h[1]:=s;g[1]:=t;
> for i from 2 to n do
> f[i]:=coeff(poly,x,i);l[i]:=f[i]-sum(g[j]*h[i-j],j=1..i-1);
> gcdex(g[0],h[0],l[i],y,'s','t');
> h[i]:=s;g[i]:=t;
> end do;
> H:=sum(h[j]*x^(j),j=0..n);
> G:=sum(g[j]*x^(j),j=0..n);
> L:=[mochar(G,n), mochar(H,n)]; RETURN(L[1],L[2]);
> end proc:
```

The routine *henselgen* takes as entry a polynomial $f(x, y)$ which is monic in y , a list L of polynomials $f_i(y)$, all monic in y , pairwise relatively prime and such that $f(0, y) = f_1(y) \cdots f_r(y)$, and an integer $n > 0$. This procedure returns polynomials $G_1(x, y), \dots, G_r(x, y)$ such that $f(x, y) = G_1(x, y) \cdots G_r(x, y)$ modulo x^{n+1} , and $G_i(0, y) = f_i(y)$. The Maple 12 code for this routine is

```

> henselgen:=proc(f,L,n)
> local P,G,H,T,S,i; S:=convert(L,set); H:=[];
> for i from 1 to nops(L) do
> T:= convert(S minus {L[i]},list);
> P[i]:=expand(product(T[k],k=1..nops(T)));
> G[i]:=Hensel(expand(f),expand(L[i]),P[i],n)[1]; H:=[op(H),G[i]];
> end do;
> RETURN(H); end proc:
```

The routine *orden* receives a polynomial “*poly*”

$$f(x, y) = y^d + b_1(x)y^{d-1} + \cdots + b_d(x),$$

with each $b_i(x)$ being a Laurent polynomial in x . It computes $u = \min\{u[i] : i = 1, \dots, d\}$, with $u[i]$ being the order of $b_i(x)$ (i.e. the degree of the least degree term in $b_i(x)$), and returns $[r, u[r]]$, with r the smallest index i such that $u[r]/r = \min\{u[i]/i\}$. Its code is

```
> orden:=proc(poly)
> local u,i,b,r,m,d;
> d:=degree(poly, y);
> b[1]:=coeff(poly,y,d-1); u[1]:=ldegree(b[1]); r:=1; m:=u[1]/r;
> for i from 2 to d do
> b[i]:=coeff(poly,y,d-i); u[i]:=ldegree(b[i]);
> if (u[i]/i < m ) then
> m:= u[i]/i; r:=i;
> end if;
> end do; RETURN([r,u[r]]);
> end proc:
```

The routine *sus* receives a polynomial “*poly*”,

$$\text{poly}(x, y) = y^d + b_1(x)y^{d-1} + \cdots + b_d(x),$$

whose coefficients are Laurent series in x and computes:

1. A polynomial $g(x, y)$ in $\mathbb{C}((x))^*[y]$ satisfying $\psi\phi(f(x, y)) = x^{du[r]}g(x, y)$, (for notation see 5) where f denotes the polynomial that is obtained from *poly* after performing the substitution $y = y - b_1(x)/d$, i.e.

$$f(x, y) = \text{poly}(x, y - b_1(x)/d) = y^d + c_2(x)y^{d-2} + \cdots + c_d(x).$$

Then r is equal to the value of the index i where the minimum value of $\{u[i]/i : u[i] = \text{degree of } c_i(x)\}$, is attained, and $u[r]$ is equal to the order of $c_r(x)$. where

$$\psi\phi : \mathbb{C}((x))^*[y] \rightarrow \mathbb{C}((x))^*[y],$$

denotes the automorphism obtained by composing the map $\phi : f(x, y) \mapsto f(x, y - b_1(x)/d)$, with $\psi : f(x, y) \mapsto f(x^r, yx^{u[r]})$.

2. *sus* returns the triple $[g, r, u[r]]$, with $g(x, y) = x^{-du[r]} \psi \phi(f/x, y)$.

Warning:

1. If $u[r] = \infty$ (which happens in case $f(x, y) = y^d$, or equivalently if $\text{poly}(x, y) = (y - b_1(x))^d$), then *sus* returns the triple $[(y - b_1(x))^d, 1, \infty]$.
2. Notice that r and $u[r]$ correspond to the polynomial $f(x, y)$ obtained from “*poly*” after performing the linear substitution that eliminates the term of degree $d - 1$ in y , but not to the polynomial “*poly*” itself.

The code for the routine *sus* is

```
> sus:=proc(poly)
> local g,q,f,b,d,u,r;
> d:=degree(poly,y);
> b[1]:=coeff(poly, y, d-1);
> f:=collect(simplify(subs( y=y-b[1]/d, poly)),y);
> r:=orden(f)[1]; u[r]:=orden(f)[2]; if u[r]=infinity then RETURN([factor(poly),r,u[r]]);end
if;
> q:=collect( expand(x ^(-d*u[r]))*subs( {x=x ^r, y=y*x ^u[r]}, f )), y);
RETURN([q,r,u[r]]);
> end proc;
```

The routine *invsus* takes as input an integer d , a polynomial $b = b(x)$, integers r, u and a polynomial $g(x, y) = y^d + c_1(x)y^{d-1} + \dots + c_d(x)$. If ψ^{-1} is the automorphism determined by $x \mapsto x^{1/r}$, $y \mapsto yx^{-u/r}$, and ϕ is the isomorphism defined as the linear substitution $y \mapsto y + b(x)/d$, then *invsus* computes $x^{du/r} \phi^{-1} \psi^{-1}(g(x, y))$.

Observation:

If $d = \text{grado}(f(x, y))$, $b = b[1]$ the coefficient of y^{d-1} in $f(x, y)$, r the smallest index i such that $u[r]/r = \min\{u[i]/i\}$, where each $u[i]$ is the degree of the coefficient of y^{d-i} in the polynomial $f(x, y - b/d)$ and $u = u[r]$ the order of its r^{th} coefficient $c_r(x)$, (quantities associated to $f(x, y)$ in the procedure *sus*, and $g(x, y)$ is a polynomial monic and of degree d in y , resulting from the previous procedure, i.e. such that $g = \text{sus}(f(x, y))[1]$, and therefore

$$g(x, y) = x^{-du[r]} \psi \phi(f/x, y).$$

Then *invsus* returns as a result $f(x, y)$, because

$$\begin{aligned} x^{du[r]/r} \phi^{-1} \psi^{-1}(x^{-du[r]} \psi \phi(f/x, y)) &= \\ x^{du[r]/r} x^{-du[r]/r} \phi^{-1} \psi^{-1}(\psi \phi(f/x, y)) &= f(x, y). \end{aligned}$$

Here is the code for *sus*.

```
> invsus:=proc(d,b,r,u,g)
> local D,t;D:=degree(g,y);
> t:=root(x,r,symbolic);
> simplify(t^(D*u)*subs({x=t, y=y*t^(-u)},g),symbolic);
> collect(simplify(subs(y=y+b/d,%)),y);
> RETURN(%);
> end proc;
```

The routine *reduccion* receives a polynomial $f(x, y)$ which is monic in y , and an integer $n > 0$. If f is linear or has the form $(y - b_1(x))^d$, the algorithm returns the triple $[1, 1, f(x, y)]$. Otherwise, it sets $g(x, y) = sus(f)$ (hence $\psi\phi(f/x, y) = x^{du[r]} g(x, y)$) and tries to verify whether $g(0, y)$ has at least one real root. If this is not the case, the algorithm returns $[1, 1, f(x, y)]$. Otherwise, it factors $g(0, y)$ in terms of the form $g(0, y) = h_1(y) \cdots h_m(y)$, where each $h_i(y)$ has the form $(y - r_i)^{d_i}$, with r_i real or of the form $h_i(y) = [(y - z)(y - \bar{z})]^{d_i}$, with z nonreal, lifts the factorization using *henselgen* modulo x^{n+1} , and applies *invsus* in order to obtain an integer $r > 0$ such that $f(x^r, y) = q_1(x, y) \cdots q_m(x, y)$. The procedure returns the list $[r, [q_1(x, y), \dots, q_m(x, y)]]$. Here is the code for this routine.

```
> reduccion:=proc(f,n)
> local i,L,S,d,h,q,r,b,g,ra,H,u;
> d:=degree(f,y); if d=1 then RETURN([1,[1,f]]); end if; b[1]:=coeff(f,y,d-1);r:=sus(f)[2];u:=sus(f)[3];if u=infinity then RETURN([1,[1,factor(f)]]); end if;
> g:=unapply(sus(f)[1],x,y);
> if [fsolve(g(0,y),y)]=[] then RETURN([1,[1,f]]); end if;
> S:=[solve(g(0,y),y)];
> L:=poli(S);
> H:=henselgen(g(x,y),L,n);
> for i from 1 to nops(H) do
> h[i]:=unapply(invsus(d,b[1],r,u,H[i]),x,y);
> q[i]:=mochar(simplify(h[i](x^r,y),symbolic),n);
> end do;
> RETURN([r,[seq(q[i],i=1..nops(H))]]);
> end proc;
```

The routine *fiscal* checks the list L and returns the empty list $[]$, if and only if all elements in L are powers of linear polynomials (i.e. each $L_i(x, y)$

in the list L has the form $(y - b_1(x))^d$, $d \geq 1$) or each $L_i(x, y)$ satisfies that $sus(L_i(x, y))$ has only nonreal roots. Otherwise, it returns the same list L . Here is the code for this routine.

```
> fiscal:=proc(L)
> local ra,g,i;
> for i from 1 to nops(L) do
> g:=unapply(sus(L[i])[1],x,y);
> ra:=[fsolve(g(0,y),y)];
> if (degree(L[i],y)>1 and ra<>[] and sus(L[i])[3]<>infinity ) then RETURN(L);
> end if; end do; RETURN([]);
> end proc:
```

The routine *cambio* receives a list $L = [r, [r_1, r_2, \dots, r_{i-1}, r_i, r_{i+1}, \dots, r_n]]$, two integers i, m , and another integer b . The procedure returns a new list $L = [br, [br_1, br_2, \dots, br_{i-1}, r_i, r_i, \dots, r_i, br_{i+1}, \dots, br_n]]$ (from the i^{th} position on it puts r_i repeated m times). The code for *cambio* is

```
> cambio:=proc(L,i,b,m)
> local n,S,h; n:=nops(L[2]);
> if i=1 then S:=[ L[1]*b, [seq(L[2][1],j=1..m),seq(b*L[2][j],j=i+1..n) ] ];
end if;
> if i =n then S:=[ L[1]*b,[ seq(b*L[2][j],j=1..n-1),seq(L[2][n],j=1..m)] ];
end if;
> S:=[ L[1]*b,[seq(b*L[2][j],j=1..i-1),seq(L[2][i],j=1..m),seq(b*L[2][j],j=i+1..n) ]];
> S; end proc:
```

The routine *factoriza* takes a polynomial $f(x, y)$ that is monic in y and an integer n . The algorithm produces a list $L = [L1, L2]$ with entries having the form $L1 = [f_1(x, y), \dots, f_n(x, y)]$ and $L2 = [r, [r_1, \dots, r_n]]$ such that $r = r_1 \cdots r_n$ and

$$f(x^r, y) = f_1(x^{r_1}, y) \cdots f_n(x^{r_n}, y),$$

such that each $f_i(x, y)$ admits no real reduction, i.e. $f_i(x, y)$ is either of the form $(y - b_1(x))^d$, with $d \geq 1$, or if $g_i(x, y) = sus(f_i(x, y))$, then $g_i(0, y)$ does not admit any real root. In other words, the only real Puiseux series in the factorization of $f(x, y)$ are the ones given by $f_1(x^{r_1/r}, y), \dots, f_n(x^{r_n/r}, y)$, and therefore if

$$f(x, y) = (y - \sigma_1(x^{1/a_1}))^{d_1} \cdots (y - \sigma_n(x^{1/a_n}))^{d_n} (y - \alpha_1(x^{1/b_1}))^{e_1} \cdots (y - \alpha_m(x^{1/b_m}))^{e_m}$$

is the complete factorization of $f(x, y)$ in $\mathbb{C}((x))^*[y]$, with $\sigma_i(t) = \sum_k p_{i_k} t^k$, a series with real coefficients p_{i_k} , and $\alpha_j(t) = \sum_k c_{j_k} t^k$, series having at least one nonreal coefficient, then $f_i(x, y) = (y - \sigma_i(x^{1/a_i}))^{d_i}$ modulo x^{n+1} , y $a_i = r_i/r$. The code for *factoriza* is

```
> factoriza:=proc(f,n)
> local FF, RR, i, redu;
> FF:=[f];RR:=[1,[1]];
> while (fiscal(FF)<>[]) do
> for i from 1 to nops(FF) do
> if reduccion(FF[i],n)[2][1]<>1 then
> redu:=reduccion(FF[i],n);RR:=cambio(RR,i,redu[1],nops(redu[2]));
> if i < nops(FF) then FF:=[op(FF[1..i-1]),op(redu[2]),op(FF[i+1..nops(FF)])];
> else FF:=[op(FF[1..i-1]),op(redu[2])]; end if;
> else end if;end do;
> end do;
> RETURN([factor(FF),RR]);
> end proc;
```

The routine *rotacion* takes a polynomial $f(x, y)$ and looks for an integer $n > 0$ such that the substitution $x = x + ny$, $y = -nx + y$ makes it quasi-monic. Afterwards it divides it by a nonzero constant making it monic. The algorithm produces a $[g/c, n]$, where g is obtained from f via the substitution, c is the coefficient of the highest power of y in the polynomial g , and n is the integer that makes this substitution work. If $f(x, y)$ is monic, the algorithm takes $n = 0$ and hence it returns $f(x, y)$ again. The code for this routine is

```
> rotacion:= proc(f)
> local c,d,g,n; for n from 0 to infinity do
> subs({x=x+n*y, y=-n*x+y},f);
> g:=collect(simplify(%),y); d:=degree(g,y);
> c:=coeff(g,y,d);
> if degree(c,x)=0 then RETURN([g/c,n]); end if;
> end do; end proc;
```

The routine *revisor* takes as entry a polynomial $f(x, y)$ that is monic in y . If $f(x, y) = g(x, y)^n$, then *revisor* return a polynomial $g(x, y)$, if $g(x, y)$ is linear in y , i.e. of the form $y - a(x)$. Otherwise, it returns the empty set. The code for this routine is

```
> revisor:=proc(f)
```

```

> local l;
> l:= factors(f)[2];
> if degree(l[1][1],y)=1 then RETURN(monico(l[1][1])); end if;
> RETURN(); end proc;

```

limite is the main routine and it is built out of the previous ones. It receives as entry polynomials $f(x, y), g(x, y)$ (not necessarily monic) and an integer $n > 0$. It computes the quotient $q = f/g$. Then it computes

$$h = y(g(\partial f/\partial x) - f(\partial g/\partial x)) - x(g(\partial f/\partial y) - f(\partial g/\partial y)).$$

Then it performs an appropriate rotation $h1 = rotacion(h)$, so that $h1$ is monic in y , and factors $h1$ into irreducible polynomials $h1 = b_1 \cdots b_s$. Next it applies *factoriza*, to an approximation of $n > 0$, and returns for each factor b_i , a list L formed by two lists

$$L1 = [[p_1(x, y), \dots, p_n(x, y)] \text{ y } L2 = [r, [r_1, \dots, r_n]]$$

such that $r = r_1 \cdots r_n$ y $bi(x^r, y) = p_1(x^{r_1}, y) \cdots p_n(x^{r_n}, y)$.

After this, it computes the list $S = [p_1(x^{r_1}, y), \dots, p_n(x^{r_n}, y)]$. Each $p_i(x^{r_i}, y) = (y - a_i(x^{r_i}))^{d_i}$, (which would correspond to a real Puiseux series) or having the form $p_i = (u(x, y))^{d_i}$, where $u(x, y)$ is monic in y , and of degree larger than one (which would correspond to a nonreal Puiseux series). Observe that $bi(x, y) = p_1(x^{r_1/r}, y) \cdots p_n(x^{r_n/r}, y)$.

After this, *revisor* is applied to S and returns a list

$$H = [q_{i1}(x, y), q_{i2}(x, y), \dots, q_{ik}(x, y)],$$

where th $q_{ij}(x, y) = (y - a_{ij}(x^{r_{ij}}))$ are exactly those elements of S , without the power d_{ij} , that correspond to real Puiseux series. That is

$$q_{i1}(x, y)^{d_{i1}} = p_{i1}(x^{r_{i1}}, y), \dots, q_{ik}(x, y)^{d_{ik}} = p_{ik}(x^{r_{ik}}, y).$$

In this way $q_{i1}(x, y) = y - a_{i1}(x^{r_{i1}}), \dots, q_{ik}(x, y) = y - a_{ik}(x^{r_{ik}})$.

Then it constructs f_1 and g_1 , the polynomial obtained by applying the substitution $x = x + ny$, $y = -nx + y$ to f and g (using the same n found for the rotation of h). After this it makes the list $T = [a_{i1}(x^{r_{i1}}), \dots, a_{ik}(x^{r_{ik}})]$.

Then it computes $R = [a_{i1}(x^{r_{i1}/r}), \dots, a_{ik}(x^{r_{ik}/r})]$, and then it computes the list P of pairs

$$P = [[f_1(x, a_{i1}(x^{r_{i1}/r})), g_1(x, a_{i1}(x^{r_{i1}/r}))], \dots, [f_1(x, a_{ik}(x^{r_{ik}/r})), g_1(x, a_{ik}(x^{r_{ik}/r}))]]$$

excluding those trajectories not passing through the origin ($a_{ik}(0^{r_{ij}/r}) \neq 0$) and therefore the list of limits

$$Q[i] = [\lim_{x \rightarrow 0} (f_1(x, a_{i1}(x^{r_{i1}/r}))/g_1(x, a_{i1}(x^{r_{i1}/r}))), \dots, \lim_{x \rightarrow 0} (f_1(x, a_{ik}(x^{r_{ik}/r}))/g_1(x, a_{ik}(x^{r_{ik}/r})))],$$

for $i = 1, \dots, s$.

The limit exists if all the values of the list $Q[i]$, for all b_i , $i = 1, \dots, s$, are equal.

```
> limite:=proc(f,g,n)
> local F,G,a,b,Q,e,P,k,T,j,N,f1,g1,i,S,H,L,h1,h,q,R,t;
> q:=f/g;
> h:= simplify(expand(y*(g*diff(f,x)-f*diff(g,x))-x*(g*diff(f,y)-f*diff(g,y))));
if h=0 then RETURN(q) end if;
> h1:=rotacion(h);
> F:=factors(h1[1])[2]; G:=[ ]; for b from 1 to nops(F) do G:=[op(G),monico(F[b][1])];
end do;
> for a from 1 to nops(G) do Q[a]:=[ ];
> L:=convert(evalf(factoriza(G[a],n)),rational); S:=[ ]; t:=root(x,L[2][1],symbolic);
> for i from 1 to nops(L[1]) do
> S:=expand([op(S),subs(x=x^L[2][2][i],L[1][i])]); end do;
> H:=map(revisor,S); N:=h1[2];
> subs({x=x+N*y, y=-N*x+y},f);f1:=collect(% ,y);subs({x=x+N*y, y=-N*x+y},g);
> g1:=collect(% ,y); T:=[ ];
> for j from 1 to nops(H) do
> T:=[op(T),-coeff(H[j],y,0)/coeff(H[j],y,1)]; end do; R:=subs(x=t,T);P:=[ ];
> for k from 1 to nops(R) do if subs(x=0,R[k])=0 then
> P:=[op(P),subs({y=R[k]},[f1,g1])]; end if; end do;
> for e from 1 to nops(P) do
> Q[a]:=[op(Q[a]),limit(evalf( P[e][1]/P[e][2] ),x=0)]; end do;
> end do; [seq(op(Q[a]),a=1..nops(F))];
> end proc;
```

4 Calculation

Next, we give some examples illustrating the performance of the algorithm:

1. $\text{limite}(6*x^3*y, 2*x^4 + y^4, 20); [2.033104508, -2.033104508, 0.]$. Consequently, $\lim_{(x,y) \rightarrow (0,0)} 6x^3y/(2x^4 + y^4)$ does not exist.
2. $\text{limite}((x^3 + y^3), (x^2 + x*y + y^2), 20); [0., 0., 0.]$. In this case $\lim_{(x,y) \rightarrow (0,0)} (x^3 + y^3)/(x^2 + xy + y^2)$ exists and equals 0.
3. $\text{limite}(6*x^3*y, 2*x^4 + y^4, 20); [2.033104508, -2.033104508, 0.]$. The limit $\lim_{(x,y) \rightarrow (0,0)} 6x^3y/(2x^4 + y^4)$ does not exist.
In the following the limit exists only in 4. and 5.
4. $\text{limite}((x^4 - y^2 + 3*x^2*y - x^2), (x^2 + y^2), 20); [-1., -1., -1.]$.
5. $\text{limite}((x^2 - y^2), (x^2 + y^2), 10); [1., -1.]$.
6. $\text{limite}((x^6 - y^4 + 3*x^2*y^3 - x^4*y), (x^4 + y^4 + x^2 + y^2), 20); [0., 0., 0., 0.]$
7. $\text{limite}(x, x^2 + y^2, 30); [\text{undefined}]$
8. $\text{limite}(y^4, x^4 + 3*y^4, 50); [0.3333333333, 0.]$
9. $\text{limite}(6*x^3*y, 2*x^4 + y^4, 10); [0., 2.033104508, -2.033104508]$.
10. $\text{limite}(x^4*y^4, (x^8 + y^8)^3, 20); [0., 0., \text{Float(infinity)}, \text{Float(infinity)}]$.

5 Discussion and Conclusions

The algorithm developed in this article provides a method for computing limits of quotients of real polynomials in two variables which has proven to be more powerful in handling these type of limits than other existing algorithms. The following examples compare the performance of the routine here presented and that of *Maple 12*.

1. Example number 5 in the previous section shows that $\lim_{(x,y) \rightarrow (0,0)} (x^4 - y^2 + 3x^2y - x^2)/(x^2 + y^2)$ is equal to -1. However, *Maple 12* is incapable to compute it.

2. Example 8 shows that $\lim_{(x,y) \rightarrow (0,0)} y^4/(x^4 + 3y^4)$ does not exist, an example that *Maple 12* computes successfully.
3. $\lim_{(x,y) \rightarrow (0,0)} (x^6 - y^4 + 3x^2y - x^4y)/(x^4 + y^4 + x^2 + y^2)$ is equal to zero, a problem that defeats Maples' routine.

The theoretical method for computing limits developed in this article applies to quotients of two real analytic functions. However, the algorithm was only implemented for polynomials. The next logical step would be to extend this algorithm to cover this general case.

In a sequel article we will develop a more general method dealing with limits of quotiens of real analytic functions in several variables.

6 Acknowledgements

We are grateful to the Universidad Nacional of Colombia and Universidad Eafit, for their invaluable support.

References

- [E] Eisenbud, D., *Commutative Algebra with a view towards Algebraic Geometry*, Springer-Verlag, 1994.G.
- [GLS] M. Greuel, C. Lossen, E. Shustin, *Introduction to Singularities and Deformations*, Springer-Verlag, 2007.
- [L] Lang, S., *Algebra*, Addison-Wesley Publishing Company Inc., 1984.
- [M] Mond, D., Saia, M., (ed.) *Real and Complex Singularities*, Lecture Notes in Pure and Applied Mathematics 232, Marcel Dekker, (2003).